LAB 7

2347210

1 Create two 3×3 matrices using the random function in Numpy and perform the following operations.

In [3]:

```
import numpy as np
```

In [28]:

```
A=np.random.randint(0,9,size=(3,3),dtype=int)
B=np.random.randint(0,9,size=(3,3),dtype=int)
print("A: ")
display(A)
print("B: ")
display(B)
```

```
A: 


array([[4, 3, 0],
       [7, 7, 2],
       [1, 7, 8]])


B: 


array([[2, 1, 0],
       [1, 4, 8],
       [6, 0, 6]])
```

1. Product (prod)

In [29]:

```
A*B
```

Out[29]:

```
array([[ 8,  3,  0],
       [ 7, 28, 16],
       [ 6,  0, 48]])
```

2. Multiplication (multiply)

In [30]:

```
np.multiply(A,B)
```

Out[30]:

```
array([[ 8,  3,  0],
       [ 7, 28, 16],
       [ 6,  0, 48]])
```

3. Dot Product (dot)

In [31]:

```
np.dot(A,B)
```

Out[31]:

```
array([[ 11,  16,  24],
       [ 33,  35,  68],
       [ 57,  29, 104]])
```

2 Perform the following set operations using the Numpy functions.

In [59]:

```
C=np.random.randint(0,9,size=(1,8),dtype=int)
D=np.random.randint(0,9,size=(1,8),dtype=int)
print("C: ",C)
print("D: ",D)
```

```
C:  [[2 5 5 1 3 6 0 4]]
D:  [[8 0 3 3 3 4 2 2]]
```

1. Union

In [60]:

```
np.union1d(C,D)
```

Out[60]:

```
array([0, 1, 2, 3, 4, 5, 6, 8])
```

2. Intersection

In [61]:

```
np.intersect1d(C,D)
```

Out[61]:

```
array([0, 2, 3, 4])
```

3. Set difference

In [63]:

```
np.setdiff1d(C,D)
```

Out[63]:

```
array([1, 5, 6])
```

4. XOR

In [64]:

```
np.setxor1d(C,D)
```

Out[64]:

```
array([1, 5, 6, 8])
```

3 Create a 1D array using Random function and perform the following operations.

In [72]:

```
E=np.random.randint(0,9,size=(1,8),dtype=int)
print("E: ", E)
```

```
E:  [[4 7 7 2 4 3 3 6]]
```

1. Cumulative sum

In [73]:

```
np.cumsum(E)
```

Out[73]:

```
array([ 4, 11, 18, 20, 24, 27, 30, 36], dtype=int32)
```

2. Cumulative Product

In [74]:

```
np.cumprod(E)
```

Out[74]:

```
array([    4,    28,   196,   392,  1568,  4704, 14112, 84672],
      dtype=int32)
```

3. Discrete difference (with n=3)

In [80]:

```
np.diff(E, n=3)
```

Out[80]:

```
array([[ -2,  12, -10,   4,   2]])
```

4. Find the unique elements from the array

In [84]:

```
F=np.unique(E)
print("F: ", F)
```

```
F:  [2 3 4 6 7]
```

4 Create two 1D array and perform the Addition using zip(), add() and user defined function (frompyfunc())

In [92]:

```
G=np.random.randint(0,9,size=(1,8),dtype=int)
H=np.random.randint(0,9,size=(1,8),dtype=int)
```

In [95]:

```
result_zip = [x + y for x, y in zip(G, H)]

result_add = np.add(G, H)

addition_func = np.frompyfunc(lambda x, y: x + y, 2, 1)
result_custom = addition_func(G, H)

print("G:", G)
print("H:", H)
print("Using zip():", result_zip)
print("Using add() function:", result_add)
print("Using custom function (frompyfunc()):", result_custom)
```

```
G: [[2 4 4 7 1 5 0 6]]
H: [[3 6 4 4 6 2 2 1]]
Using zip(): [array([ 5, 10,  8, 11,  7,  7,  2,  7])]
Using add() function: [[ 5 10  8 11  7  7  2  7]]
Using custom function (frompyfunc()): [[5 10 8 11 7 7 2 7]]
```

5 Find the LCM (Least Common Multiple) and GCD (Greatest Common Divisor) of an array of elements using reduce().

In [99]:

```
from functools import reduce
import math

def lcm(x, y):
    return x * y // math.gcd(x, y)

def gcd(x, y):
    return math.gcd(x, y)

arr = [12, 18, 24, 36]

lcm_result = reduce(lcm, arr)

gcd_result = reduce(gcd, arr)

print("Array:", arr)
print("LCM of the array elements:", lcm_result)
print("GCD of the array elements:", gcd_result)
```

```
Array: [12, 18, 24, 36]
LCM of the array elements: 72
GCD of the array elements: 6
```